Password Encryption Key

CLAIM OF PRIORITY

This application claims priority under 35 USC 119(e) to U.S. Patent Application Serial No. 60/421,284 filed on October 25, 2002, the entire contents of which are hereby incorporated by reference.

TECHNICAL FIELD

This invention relates to eSecurity and more particularly to user authentication.

BACKGROUND

A frequently neglected aspect of the modern enterprise data storage is sensitive user information security. The most widespread approach used today is encryption of such user information as Social Security number, credit card numbers, e-mails, etc. with a single key and storage of the resulting encrypted data in the database. The logic behind such solution is that a malicious individual who gains access to the database will be unable to make use of the user's sensitive data because it is encrypted.

Unfortunately, this approach provides a false sense of security in most cases. The problem is that the encryption key used to encrypt all records still needs to be stored somewhere in the system. For example, as soon as the system is required to send e-mail to the user or submit user's credit card number to the merchant account, the server(s) responsible for fulfilling that requirement must use the key to decrypt user information retrieved from the database. Chances are that if a malicious individual manages to get access to the database, which is usually the most protected part of the system, he will then be able to gain access to the aforementioned server. As soon as this happens, such malicious individual will be able to obtain the key and decrypt every database record encrypted with this key.

SUMMARY

A password-encrypted key (PEK) is generated from a user-supplied password, for example, and then used to encrypt the user's password. The encrypted password is stored in a user record on a server. At login, a would-be user's password is again used to make a key, which is then used to decrypt and compare the stored encrypted password with the would-be user's password to complete the login. The successful PEK is stored in a temporary session record and can be used to decrypt other sensitive user information previously encrypted and stored in the user record as well as to encrypt new information for storage in the user record. A public/private key system can also be used to maintain limited access for the host to certain information in the user record.

According to one aspect of the invention, a secure transaction process includes generating a key from a user-supplied unencrypted password or other identifying data, encrypting the user's password with the key, creating a user record and storing the encrypted password in the user record. In another aspect of the invention, upon user login, a key is made from a would-be user's password using the same algorithm used to generate the key from the originally supplied unencrypted password, then the encrypted password in the corresponding user record is retrieved and decrypted using the key and the decrypted password and the would-be user-supplied password are compared to see if they match.

In the preferred process, if the decrypted password and user-supplied password match, a temporary session record is created and the key is stored in the session record. In the absence of a match, the user login procedure for secure or user-authenticated transactions at least would preferably be aborted or terminated in some fashion.

The key may be used to encrypt other sensitive user data, which can be stored in the user record. During a session in which a session record has been created, the key stored in the session record can be used to decrypt the other encrypted information stored in the user record for use in carrying out some desired action.

Alternatively, a public/private key pair or other asymmetric cryptography can be employed. A public/private key pair is generated and the public key is stored on an application server and the mating private key only on another server, preferably a secure off-site server. The original user-supplied unencrypted password is then encrypted with the public key and stored on the application server. Subsequently, the private key can be fetched from the other server and used to decrypt selected information on the one server, for example, for a mass mailing. A single public/private key can support the entire site.

The password encryption key (PEK) system of the present invention eliminates one of the shortcomings of prior methods by using a unique encryption key for each user record. This key is based on at least one piece of data – user password. Optionally, user name (or user ID) can be used in conjunction with user password. User's password (and name) is obtained at each successful user login and is maintained by the system for the duration of that user's session.

The details of one or more embodiments of the invention are set forth in the accompanying drawings and the description below. Other features, objects, and advantages of the invention will be apparent from the description and drawings, and from the claims.

The details of one or more embodiments of the invention are set forth in the accompanying drawings and the description below. Other features, objects, and advantages of the invention will be apparent from the description and drawings, and from the claims.

DESCRIPTION OF DRAWINGS

FIG. 1 is a flowchart showing the user registration process.

FIG. 2 is a data structure diagram of the user record.

FIG. 3 is a flowchart showing the user login process.

FIG. 4 is a data structure diagram of the session record.

FIG. 5 is a flowchart showing a process for safely storing sensitive information.

FIG. 6 is a flowchart showing the process for safely retrieving and using stored sensitive information, which has been encrypted.

FIG. 7 is a flowchart showing an alternative registration process using a public key.

FIG 8 is a flowchart showing the process for extracting e-mail addresses using public/private key pairs stored in the process of FIG 7.

Like reference symbols in the various drawings indicate like elements.

DETAILED DESCRIPTION

PEK integration into the system

The PEK system is illustrated as a sequence of processes as shown in FIGS. 1, 3, 5 and 6, running on an application server or other computer system. Preferably all of the processes are carried on the Internet on a server that hosts a given application accessed remotely by a user from his or her personal computer, for example.

1. User Registration Process (FIG. 1)

a. User registers by, at least, providing new username and an arbitrary password.

b. Generate a key from the password. (and optionally username/ID). A key can be generated by calculating an MD5 checksum of the source data.

c. Encrypt user password with the key obtained at step 1b. ***Note:*** other sensitive data provided during registration (i.e. e-mail address) should also be encrypted with the same key.

d. Create new user record (FIG. 2) and store username, encrypted password and other optional data (sensitive data encrypted) in that user record.

2. User Login Process (FIG. 3)

a. User logs in providing username/password for authentication.

b. Generate a key from the password (and optionally username/ID). This key should be identical to the key obtained at step 1b.

c. Retrieve user record by username and decrypt user password using the key obtained at step 2b.

d. Compare the decrypted password to the one provided by user at step 2a.

e. Reject user login if passwords do not match. Abort login process.

f. If passwords match, create a temporary user session record (FIG. 4) that will exist for the duration of the user session. (A user session is a temporary data pool created after user login and destroyed as a result of explicit user logout or session timeout. Session timeout occurs after a certain pre-determined period of user inactivity.)

g. Store resulting key in the session.

h. Communicate session ID back to the user (client application). Session ID is a number or string uniquely identifying the session. Once user (client application) receives the session ID from the system, user will always provide that ID with each subsequent request for the duration of the session. This enables the system to get access to user session data at each request.

3. Sensitive Information Storage (FIG. 5)

a. User submits some sensitive data (i.e. Credit Card number).

b. Encrypt sensitive data with a key retrieved from user's session.

c. Store encrypted data in user's record if it is to be permanently maintained on the server. If it is only to be available for the session then the encrypted data would be stored only temporarily in the session record.

4. Sensitive Information Retrieval (FIG. 6)

a. User requests some system action requiring use of the information stored at step 3. (i.e. user decides to make a purchase with the credit card that he/she previously submitted to the system).

b. Retrieve a key (i.e., the PEK) from the user's session record (FIG. 4).

c. Decrypt the necessary data using the key obtained at step 4b.

d. Perform the required action with decrypted data (i.e. send it to the merchant account)

e. Discard decrypted data.

Implications

The system, at user's request, can decrypt data stored in the database at step 3 (FIG. 5). at any time while that user's session is active. As soon as user's session expires, it should be impossible to decrypt this user's sensitive information without knowing the user's password. Note that user's password is also encrypted at step 1c (FIG. 1).

Thus, a user's sensitive data will always be as secure as the user's password in this system. In the majority of cases, this should be acceptable since knowledge of password only gives access to sensitive user account information through the standard interface anyway.

Potential Vulnerabilities and Solutions

While PEK offers a secure way of protecting user data for users that are not currently logged in, in theory, it could be possible for a malicious individual to gain access to sensitive data for users that are currently logged in (i.e. have active sessions going). Such individual would have to obtain all of the encrypted user data and all of the active sessions data, extract a key from each session, and decrypt the active user's sensitive data by applying extracted keys to corresponding user records.

Logged in users, however, in most cases, represent only a small subset of all registered users and that alone greatly limits the scope of potential risks. In addition, the exposure can be further limited by making sure that the information linking session to a specific user, like username/ID, is not stored in session data. Instead, this information can be provided by the client application with each user's request. That alone would make it exceedingly difficult for a malicious individual to match a key, retrieved from any given session, to a specific user record.

Other Considerations

PEK makes it difficult to perform legitimate system functions that involve access to sensitive user data without an explicit user request. Bulk mailing to all system users is a good example. Suppose that user e-mails or at least e-mail addresses are encrypted using PEK. It will then be impossible for the system to decrypt user e-mails because each e-mail is encrypted with its owner's password and that password itself is also encrypted.

One solution to this problem is to utilize asymmetric cryptography, like PGP, and keep a copy of the user's password, encrypted with a public key, in the main database, as shown in FIG. 7. Only one pair of public/private keys for the entire site needs to be generated in advance; then the public key, used for encryption, should be stored on the application server, while the private key, used for decryption, should be stored on an off-site secure server. This server, at the time of bulk mailing, as shown in FIG. 8, will decrypt user's password using that private key, generate user's PEK as described in step 1b, and, finally, decrypt required information using PEK. The main advantage of this approach is that it should be possible to keep the server, which maintains the private key, either off-site or in a special security zone. This setup will ensure that while this server will be able to access the system data, the system would not be able to access the server. To further enhance security, this server can also be completely inaccessible (i.e. down) when bulk operations are not in progress.

Another approach is to use the public key to directly encrypt only those user record fields that require bulk access. Distinct public/private key pairs can be used to encrypt different field types (i.e. e-mails and Credit Card numbers). This would allow for a more refined access permissions control. For example, bulk mail server will only have a private key that decrypts e-mails, but not Credit Card numbers.

Finally, yet another approach could be to push unencrypted data to the off-site server at the time of its submission by user. This is the least secure approach but it allows the most flexibility.

A number of embodiments of the invention have been described. Nevertheless, it will be understood that various modifications may be made without departing from the spirit and scope of the invention. For example, instead of MD5 checksum, some other encryption algorithm or reproducible key-making methodology could be used. Accordingly, other embodiments are within the scope of the following claims.